Fast Summation Boundary Element Method for Calculating Solvation Free Energies of Macromolecules

ENRICO O. PURISIMA

Biotechnology Research Institute, National Research Council of Canada, 6100 Royalmount Avenue, Montreal, Quebec H4P 2R2, Canada

Received 24 October 1997; accepted 6 May 1998

ABSTRACT: We present a boundary element method (BEM) for calculating the reaction field energy of a macromolecule embedded in a high-dielectric medium such as water. In a BEM calculation, the key computational task is the calculation of the induced surface charge distribution at the dielectric boundary. This is obtained by solving a system of linear equations whose dimension can run into the tens of thousands for a macromolecule. In this work, we use a fast summation hierarchical multipole method to solve for the induced surface charge densities. By careful analysis of the levels of approximation required for the various terms in the calculation, we avoid the unnecessary computation of terms that contribute negligibly to the final outcome and, consequently, achieve high computational efficiency. For a protein such as BPTI with 890 atoms, the calculation of the induced surface charge density distribution and the reaction field energy was completed in 7.9 s on an SGI workstation with an R10000 CPU.
 J Comput Chem 19: 1494–1504, 1998

Keywords: continuum dielectric; hierarchical multipole; boundary element; solvation

Introduction

Electrostatic interactions and solvation effects play a major role in the properties and activities of biological molecules.[1,2] Computer simulation studies of conformational stabilities, binding affinities, pK shifts require the proper incorporation of solvation effects. However, simulations involving systems solvated by explicit water molecules can be extremely expensive especially for macromolecules such as proteins. Continuum dielectric approximations are a popular inexpensive alternative for incorporating solvation effects.[1–4] In this approximation, the solute is treated as a low-dielectric cavity embedded in a high-dielectric medium representing the solvent. The Poisson or Poisson–Boltzmann equation is then solved nu-

Correspondence to: E. O. Purisima; e-mail: rico@bri.nrc.ca
Contract/grant sponsor: National Research Council of Canada; publication number: 41432

 CCC 0192-8651 / 98 / 131494-11

merically. The numerical solution is most commonly carried out using a finite-difference method in which the electrostatic potential is solved on a three-dimensional cubic grid enclosing the solute charges.[5–7] An alternative solution to the Poisson equation is the use of boundary element methods (BEM).[8–10] An advantage of BEM is that solvation free energies are more readily calculated as compared with finite-difference methods where one requires special treatment for the self-energies of the solute charges.

In the boundary element method, the surface defining the dielectric interface is tessellated and an induced surface charge distribution is calculated. Then the reaction field energy can be readily obtained. The technical difficulty in this method lies in calculating the surface charge distribution. The process involves the solution of an n by n system of linear equations where n is the number of surface boundary elements. For a macromolecule, n can be in the tens of thousands. This poses severe storage and computation time costs. Recent approaches in surmounting this problem have made use of multipole expansions and fast summation approaches[11,12] as well as multigrid methods.[13] These have reduced computation costs significantly, but there is still much room for improvement for calculations on macromolecules. In this article, we present a fast summation method that significantly reduces the computational cost of BEM calculations and makes it competitive with finite-difference methods, even for macromolecular systems such as proteins.

By carefully analyzing what levels of approximation are required for the various terms, we avoid unnecessary computation of terms that contribute negligibly to the final outcome. We show that a monopole approximation is sufficient for the off-diagonal terms in the surface charge interaction matrix, whereas a first order approximation is needed for the diagonal terms. By calculating the diagonal terms implicitly, and by judicious choice of approximation level, we have been able to reduce significantly the computational cost of the BEM. The algorithm is implemented as a C++ program called BRI BEM (*B*iotechnology *R*esearch *I*nstitute *B*oundary *E*lement *M*ethod).

Method

The reaction field energy of a solute molecule immersed in a medium of different dielectric can be calculated as

$$G_{rf} = \frac{1}{2}\sum_k q_k \int_S \frac{\sigma(\mathbf{r})}{|\mathbf{r} - \mathbf{r}_k|}\, dS \tag{1}$$

where σ is the induced surface charge density and q_k and $\mathbf{r}_k$ are the partial charge and position of atom i, respectively. In the boundary element method, we represent the surface by a discrete mesh with an effective uniform charge density within each surface element. The reaction field energy can then be approximated as

$$G_{rf} = \frac{1}{2}\sum_k \sum_j q_k \frac{\sigma_j A_j}{|\mathbf{r}_j - \mathbf{r}_k|} \tag{2}$$

where σ_j and A_j are the charge density and area of patch j, respectively. The values for σ_j are obtained by solving the system of linear equations[8–10]

$$(\mathbf{I} - f\mathbf{K})[\sigma] = f[E] \tag{3}$$

where

$$f = \frac{1}{2\pi}\left(\frac{D_{\text{in}} - D_{\text{out}}}{D_{\text{in}} + D_{\text{out}}}\right) \tag{4}$$

D_{in} and D_{out} are the dielectric constants in the solute and solvent, respectively. $\mathbf{I}$ is the n by n identity matrix, $[\sigma]$ the column vector of patch charge densities, and $[E]$ the column vector of the normal component of the electric field at each of the patch centers due to the solute charge distribution

$$E_j = \frac{1}{D_{\text{in}}}\sum_k \frac{(\mathbf{r}_j - \mathbf{r}_k)\cdot \mathbf{n}_j}{r_{jk}^3} q_k \tag{5}$$

The elements of the matrix $\mathbf{K}$ depends only on the geometry of the dielectric interface and the mesh used to define it. The off-diagonal elements of the matrix $\mathbf{K}$ are computed as[8–10,14]

$$K_{ij} = \frac{(\mathbf{r}_i - \mathbf{r}_j)\cdot \mathbf{n}_i}{r_{ij}^3} A_j \tag{6}$$

Eq. (6) is obviously not valid for the diagonal elements where $i = j$. We have shown previously that we can express the diagonal elements as a linear combination of the off-diagonal ones[14]

$$K_{ii} = 2\pi - \sum_{j\neq i} K_{ji}\frac{A_j}{A_i} \tag{7}$$

This sum rule is not valid at singular points on the surface (e.g., cusps). Thus, the molecular surface programs that we use in our work filter out singularities from the molecular surface prior to the boundary element calculation.

Multipole expansion. In applying multipole methods to the solution of eq. (3), we build upon the formulation of Zauhar and Varnek.[11] They decomposed $\mathbf{K}$ into two components, $\mathbf{K}_F$ and $\mathbf{K}_N$

$$(\mathbf{I} - f\mathbf{K}_N)[\sigma] = f([E] + \mathbf{K}_F[\sigma]) \qquad (8)$$

where $\mathbf{K}_N$ and $\mathbf{K}_F$ contain those K_{ij} values corresponding to near and far interactions, respectively, based on the distance, r_{ij}, between points i and j. The right-hand side of eq. (8) is an n-dimensional vector whose elements are the sum of the normal components of the electric field at each surface point due to the solute charges and due to the surface charge density from faraway patches. $[E]$ is calculated once, and $\mathbf{K}_F[\sigma]$ can be rapidly calculated using multipole approximations.[11] The idea is to make the left-hand side of eq. (8) a sparse matrix affording rapid solution of the system of equations as well as significantly reducing memory costs.

From eq. (6), the ith element of the vector $\mathbf{K}_F[\sigma]$ is

$$(\mathbf{K}_F[\sigma])_i = \left(\sum_{\text{far } j} \frac{\mathbf{r}_i - \mathbf{r}_j}{r_{ij}^3} \sigma_j A_j \right) \cdot \mathbf{n}_i \qquad (9)$$

It is the summation within parentheses on the right-hand side of eq. (9) that is computed via multipole approximation. In a multipole approximation, a cubic grid of cells is superimposed on the system and points are assigned to their respective cells. The electric field due to faraway cells can then be approximated by monopole, dipole, and quadrupole contributions from the centers of the cells. Eqs. (17)–(19) of Zauhar and Varnek[11] give the standard multipole expressions in cartesian coordinates. For example, the multipole approximation of the field of cell ν felt at surface element i is shown below with the monopole term shown explicitly

$$\mathbf{E}_\nu(\mathbf{r}_i) \approx q_\nu \frac{\mathbf{r}_i - \mathbf{r}_\nu}{r_{i\nu}^3} + \text{dipole} + \text{quadrupole} \qquad (10)$$

$$q_\nu = \sum_{k \in \nu} \sigma_k A_k \qquad (11)$$

where $\mathbf{r}_\nu$ is the center of cell ν. In this work, near cells are the immediate neighbors of the current cell. All other cells are considered faraway cells.

A further approximation that can further reduce computational cost can be obtained by using a hierarchical multipole approximation and local Taylor expansion.[15] In this approach, a multilevel cubic grid is defined. The lowest useful level is a $4 \times 4 \times 4$ grid with higher levels being $2^n \times 2^n \times 2^n$ grids. In going from one level to the next, each cell is subdivided into eight child cells by halving the grid spacing along each axis. The idea in this multilevel approach is that rather than carrying out a point–cell interaction calculation as in eq. (10), we calculate a cell–cell interaction at the lowest (coarsest) level and iteratively propagate the interaction to child cells at the higher levels via Taylor expansions. Only at the highest level is the field at individual surface points in the cell calculated. It is calculated via Taylor expansions about the center of the cell. The use of a hierarchical multipole approach has been described by Bharadwaj et al.[12] Detailed descriptions of the fast multipole method, local Taylor expansions and hierarchy of cells applied to the calculation of long-range nonbonded interactions in molecules can be found in the literature[16–19] and are based on the work of Greengard and Rokhlin.[15]

The local Taylor expansion method is illustrated schematically in two dimensions in Figure 1. Three levels of subdivision are shown. C2 is the center of a level 2 cell highlighted as a darkly outlined square. The cells more than one cell unit away are labeled ''2'' and are considered far cells. The electric field of these cells at the C2 is calculated from their multipole moments. Then, using a local Taylor expansion, the field from the ''2'' cells at C3 is obtained. C3 is the center of one of the child cells of the original cell. At this next higher subdivision, the far cells whose fields have not yet been taken into account are labeled ''3.'' Their contribution to the electric field at C3 can also be calculated from their multipole moments and are added to the field C3 has ''inherited'' from its parent cell. Then we go to the next level, generate child cells and use a Taylor expansion about C3 to calculate the field that C4 inherits. To this is added the far-field from the ''4'' cells. Finally, the total far field felt at each of the surface elements contained in the cell of C4 is calculated via a Taylor expansion about C4. A detailed description of the Taylor expansion formulas used in this work for the calculation of $\mathbf{K}_F[\sigma]$ is given in the Appendix.

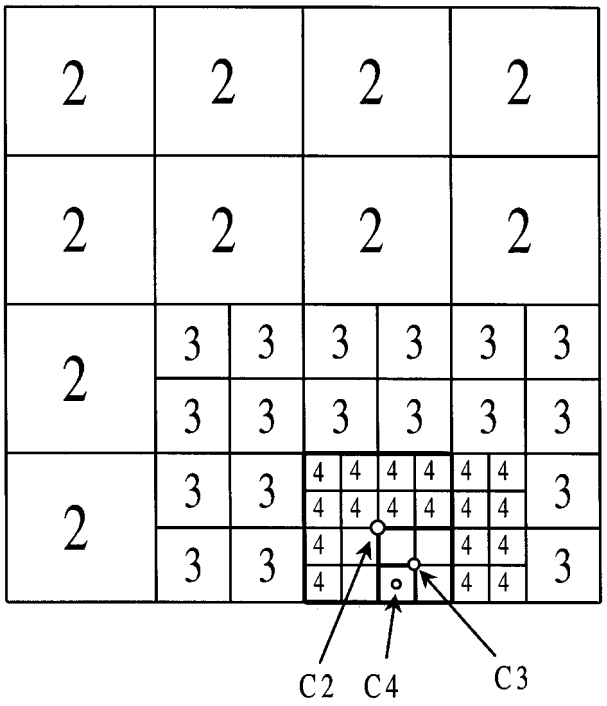

FIGURE 1. Schematic diagram of the hierarchical grid used for the multipole and local Taylor expansion calculation. See text for discussion.

The calculation of K_{ii} from eq. (7) is also an n^2 calculation that can likewise benefit from a fast summation treatment. Eq. (7) can be written out more explicitly as

$$K_{ii} = 2\pi - \sum_{j \neq i} \frac{\mathbf{r}_j - \mathbf{r}_i}{r_{ij}^3} \cdot \mathbf{n}_j A_j \qquad (12)$$

Unlike in eq. (9), the unit normal cannot be factored out of the summation. Nevertheless, it is still possible to derive a fast summation expression for this sum. The details are described in the Appendix.

Results

Our main goal in this study is to determine the minimum number of terms required in the formulas for the various multipole and Taylor expansions to achieve an acceptable level of accuracy. As mentioned in the ''Methods'' section, the linear system, eqs. (3) or (8), is solved by Gauss–Seidel iterations. The first issue then was to define our convergence criterion.

Convergence criterion. There is no unique choice for defining convergence in the Gauss–Seidel iterations. Zauhar and Varnek[11] and Bharadwaj et al.[12] chose to monitor quantities related to the relative difference in the surface charges from iteration to iteration. When the change between iterations in their defined quantities fell below a given tolerance, convergence was reached. In this work, we chose the relative change in reaction field energy between iterations as the quantity to monitor. This seems appropriate because it is this energy that is the main physical quantity of interest from the calculation. At each iteration, the reaction field energy was calculated with eq. (2) using the values of $[\sigma]$ for that iteration. Convergence was reached when $|G_{i+1} - G_i|/|G_i| < \text{tol}$.

We tested the effect of the convergence tolerance on the final value of the reaction field energy. Tolerances in the range of 10^{-1} to 10^{-8} were tested. Three systems were used for the test: a 9-Å sphere with an off-center unit charge, a nine-residue alanine polypeptide in an alpha helix, and the small protein crambin with 653 atoms. Partial charges and atomic radii for the helix and crambin atoms were taken from the AMBER force field[20] with the radii of polar hydrogens modified to be 1.0 Å. The molecular surfaces were generated using a marching tetrahedra approach.[21,22] All calculations were carried out using the full matrix; that is, with no multipole approximation. The solute and solvent dielectric constants were 2 and 78.5, respectively.

The results are summarized in Table I. We see that the final value of the reaction field energy no longer changes significantly below the 10^{-5} tolerance. For all our subsequent calculations then, we chose 10^{-5} as our convergence tolerance. In actual practice, we could use a less stringent convergence tolerance and still obtain reasonable results with a reduced computational cost. For the next subsections we examine the effects of introducing multipole or Taylor series approximations on accuracy and computing time. Due to the importance of the diagonal terms, we investigated the effects of approximating the diagonal and off-diagonal contributions separately. For the following calculations, we used crambin as our test molecule.

Off-diagonal terms. The effect of a multipole approximation for the off-diagonal terms, that is, the use of eqs. (9) and (10), was examined at the monopole, dipole, and quadrupole levels. The values for $(\mathbf{K}_F[\sigma])_i$ were updated at every other Gauss–Seidel iteration. The diagonal terms were calculated using the full matrix via eq. (7). Two variations in using the grid were tried. In one case,

TABLE I.
Reaction Field Energies and Number of Iterations as a Function of Convergence Tolerance.

Tol	Sphere	Helix	Crambin
10^{-1}	−18.450307 (3)	−12.173894 (3)	−113.782372 (2)
10^{-2}	−18.435985 (4)	−12.171216 (4)	−115.469092 (4)
10^{-3}	−18.435985 (4)	−12.171216 (4)	−115.494674 (5)
10^{-4}	−18.435657 (7)	−12.169567 (5)	−115.493430 (6)
10^{-5}	−18.435655 (9)	−12.169929 (6)	−115.493831 (7)
10^{-6}	−18.435654 (10)	−12.169901 (7)	−115.493809 (8)
10^{-7}	−18.435654 (12)	−12.169901 (7)	−115.493823 (10)
10^{-8}	−18.435654 (13)	−12.169908 (12)	−115.493823 (10)

All energies are in kcal / mol. The sphere has a 9-Å radius and a unit charge displaced 3 Å from the center. The helix is a nine-residue alanine polypeptide. The sphere, helix, and crambin are represented by 960, 4481, and 17,495 surface elements, respectively. These correspond to densities of 0.94, 7.2, and 7.0 points / Å^2, respectively. In parentheses are the number of Gauss–Seidel iterations needed to converge to within the tolerance given in the first column. Convergence is reached when the magnitude of the relative difference in reaction field energies between two successive Gauss–Seidel iterations is less than the tolerance value. Calculations have been carried out without the multipole approximation; that is, using the full matrix in eq. (3).

the centers of the cells were simply taken as the geometric center of that cube. In the second case, we used a weighted cell center defined as the average position of the points in the cell computed as

$$\text{Cell center} = \sum_{j \in \text{cell}} \mathbf{r}_j A_j \Big/ \sum_{j \in \text{cell}} A_j \qquad (13)$$

The motivation for the second method was to reduce the magnitude of higher order moments associated with the cell. The results for our sphere, helix, and protein test cases are summarized in Table II. The results should be compared with those for the full matrix calculation (Table I). The CPU time is broken down into the major computational components: (a) computation of the surface charge distribution [i.e., solution of eq. (8)]; (b) calculation of the components of $[E]$ via eq. (5); (c) calculation of the diagonal elements via eq. (7); and (d) other miscellaneous setup calculations.

What we see is that for the off-diagonal elements, a monopole approximation gives sufficient accuracy. This is fortunate because the off-diagonal elements form the bulk of the calculation and the savings in not having to use higher order terms is significant. The use of weighted cell centers seems to improve the accuracy of the monopole approximation. The results for the dipole and

TABLE II.
Energies Using Multipole Expansion of Off-Diagonal Terms for Crambin.

			CPU time (seconds)			
	Energy	No. of iterations	Calc. σ	E_{norm}	K_{ii}	Setup
Uniform cell centers						
Monopole	−115.85	14	25.38	4.24	171.83	6.67
Dipole	−115.53	14	29.31	4.22	172.37	6.68
Quadrupole	−115.53	14	57.44	4.23	179.18	6.71
Weighted cell centers						
Monopole	−115.61	14	27.40	4.25	178.31	6.98
Dipole	−115.44	14	31.37	4.24	179.54	6.95
Quadrupole	−115.55	14	60.09	4.24	179.20	6.96

All energies are in kcal / mol. The diagonal terms were calculated using eq. (7), requiring evaluation of the elements of the full matrix. The contribution of the off-diagonal elements was calculated using eq. (9). The CPU times are in seconds and are for a 195-MHz SGI-R10000 CPU. The crambin surface is as in Table I. A grid spacing of 1.1 Å for the highest level subdivision was used in the multipole expansion. This results in an average of about 7 surface points / cell at the finest grid level for nonempty cells.

TABLE III.
Reaction Field Energies Using a Monopole Approximation for Off-Diagonal and First-Order Expansion of Diagonal Terms.

			CPU time (seconds)			
	Energy	No. of iterations	Calc. σ	E_{norm}	K_{ii}	Setup
Uniform	−115.69	14	25.84	4.25	4.45	6.60
Weighted	−115.50	14	27.18	4.24	4.67	6.95

K_{ii} calculated using eq. (A2). Energy expressed in kcal / mol.

quadrupole approximations are not much affected by the choice of cell center. We note that the solution of eq. (8) and calculation of the diagonal are the most CPU-intensive.

Diagonal terms. The next step was then to use the monopole approximation for the off-diagonal terms and use the first-order approximation for the diagonal terms using eq. (A27). Both the uniform and weighted cell centers were tested. The results are summarized in Table III. Again, the use of weighted cell centers gives more accurate results. The calculation time for K_{ii} is reduced by a factor of 40. This is because the direct calculation of the diagonal terms using eq. (7) is an order n^2 calculation whereas the use of eq. (A27) is an order $n \log n$ calculation. At this point, the solution of eq. (8) is the most costly part of the calculation.

Local Taylor series expansion. To improve performance even further, a local Taylor series expansion was used for the far-field interactions of the off-diagonal terms, as described in the ''Method'' section and Appendix. Because the monopole approximation for the far field was shown to be sufficient, a local Taylor series expansion of this monopole interaction referred to the centers of the cells was used. The results are summarized in Table IV. We see that there is negligible loss of accuracy while improving performance another three- to fourfold in the calculation of σ. Total CPU time for the calculation of the surface charge densities and reaction field energy for crambin has now been reduced to under 17 seconds.

Discussion

The large number of surface elements and the consequent large matrix size in eq. (3) has been the main deterrent in the use of BEM calculations for macromolecular systems such as proteins. Zauhar and Varnek[11] and Bharadwaj et al.[12] have described methods using multipole approximations to address this problem. In their work, they report significant improvement in speed over a full matrix calculation. However, the performance in their BEM calculations still does not approach that of the more popular finite-difference calculations.[6,7]

The method described in this work is a fusion of the approaches of Zauhar and Varnek[11] and Bharadwaj et al.[12] in the context of the sum-rule calculation of Purisima and Nilar.[14] The speed of the current algorithm arises from the combined effects of several factors:

1. The use of a hierarchical fast multipole and local Taylor series[18] evaluation of $\mathbf{K}_F[\sigma]$ in eq. (8) enhances performance over the original formulation of Zauhar and Varnek.[11]

TABLE IV.
Reaction Field Energies Using Local Taylor Expansion Approximation.

			CPU time (seconds)			
Expansion	Energy	No. of iterations	Calc. σ	E_{norm}	K_{ii}	Setup time
$\mathbf{r} / r^3$	−115.53	14	5.95	4.23	4.23	2.16
$1 / r^3$	−115.55	14	6.03	4.23	4.24	2.16

Two values for crambin are shown. The first uses the Taylor expansion of $\mathbf{r} / r^3$ as in eq. (A2); The second uses a Taylor expansion of $1 / r^3$ and the exact value of $\mathbf{r}$. There is little difference between the two results.

2. The use of Cartesian coordinates instead of spherical harmonics[12] for the multipole expansions reduces computational costs by avoiding the calculation of transcendental functions.
3. The use of a monopole approximation for the far-field off-diagonal interactions affords considerable savings in time. Bharadwaj et al.[12] included terms up to the octupole level in their multipole expansion, thus incurring significant computational costs.
4. The use of the sum rule, eq. (7), improves the self-consistency of the matrix elements, K_{ij}, and may enhance the convergence of the Gauss–Seidel iterations. The application of the sum rule probably also mitigates the errors in using the monopole approximation for the far-field off-diagonal interactions.
5. The boundary element method can be sensitive to the quality of the surface mesh used to describe the molecular surface. In this work, we used either a marching tetrahedra procedure[21,22] or, for the tests described in what follows, the SIMS program (*S*mooth *I*nvariant *M*olecular *S*urface) of Vorobjev and Hermans.[23] These provide a generally smooth well-behaved mesh that assists in the convergence of the Gauss–Seidel calculations.

Comparison with a multigrid method. Vorobjev and Scheraga[13] use a different approach to solve eq. (3). They use a multigrid boundary element method. Three sets of boundary elements of varying levels of coarseness are defined on the molecular surface. To each charge center in the solute they associate a local surface represented by small boundary elements, an intermediate surface represented by large elements, and a distant region represented by even larger elements that they call patches. The idea is that for *each* charge center, they are able to rewrite eq. (3) as a system of linear equations of low dimension. Each charge center has its own system of linear equations to be solved. Using this method, they showed that the time required to calculate the reaction field energy of a 17-residue helical polypeptide on one node of an IBM-SP2 supercomputer was comparable to that required for a DelPhi calculation with lattice dimensions $N_g = 129$ and lattice size $h = 0.227$ Å.[13] Thus, their method is competitive with the speed of finite-difference calculations for this size of molecule.

As suggested by a reviewer, we carried out a side-by-side comparison of the method of Vorobjev and Scheraga with our method. Their program FAMBE (*F*ast *A*daptive *M*ultigrid *B*oundary *E*lement method) was downloaded (see ''Acknowledgments'') along with a PDB file containing the coordinates, radii, and atomic partial charges of the 17-residue peptide used in their study. To make a proper comparison, we needed to use the same surface points. Thus, we also downloaded SIMS,[23] which is the surface generation program built into the current version of FAMBE. SIMS provides a surface free of singularities commonly found in exact Connolly surfaces.[24]

Reaction field energies were calculated at surface densities of 1, 2, 4, 8, and 16 points/Å^2. The results are summarized in Table V. We see that our method is approximately three times faster than FAMBE in this test case. It is reassuring that

TABLE V.
Comparison with FAMBE.

	FAMBE		BRI BEM		
Density	Energy	CPU(seconds)	Energy	CPU(seconds)	Grid spacing
1	−196.23	4.35	−191.43	1.24	2.4
2	−197.66	7.03	−199.18	2.45	1.9
4	−197.29	13.47	−202.78	5.24	1.5
8	−201.51	33.58	−202.05	11.94	1.0
16			−202.36	27.97	0.7

FAMBE is the boundary element method of Vorobjev and Scheraga.[13] The calculations were carried out on a 17-residue test peptide, Ac-ETGTKAELLAKYEATHK-Nme, used in their study. Solute and solvent dielectrics were set to 1 and 80, respectively. Surface points were generated using SIMS[23] at the nominal densities indicated (in points/Å^2). There is no value for FAMBE at 16 points/Å^2, because the version of FAMBE available to us could not handle densities greater than 10 points/Å^2. The total surface area is about 1400 Å^2. The grid spacing shown for BRI BEM is the cell dimension at the highest (finest) level in the hierarchical multipole calculation, and was chosen to maintain an occupancy of 5–8 surface points per nonempty cell

the two very different methods give very similar values for the calculated reaction field energies. One advantage of FAMBE that we observed was its robustness to imperfections in the molecular surface. Although SIMS filters out singularities, it still does occasionally produce ''pinched'' surface regions that are not singular, but where two nonadjacent surface patches can come very close. An example would be two concave regions that almost intersect. FAMBE does not seem to be affected adversely by the presence of such regions. On the other hand, to achieve good convergence in our method, we had to write a small utility program that automatically filters out points belonging to such regions.

Standard parameters for production runs. It is useful at this point to define a standard set of parameters to use for BRI BEM in production runs. With respect to the multipole calculation, a monopole approximation for off-diagonal terms with a local Taylor series expansion gives the best performance with negligible loss in accuracy. The diagonal terms are calculated using a first-order approximation. Near cells are defined as the immediate neighboring cells; all other cells are far and treated using the multipole approximation.

In our method, the appropriate value of the grid size depends on the density of the surface points. As in applications of the multipole method to molecular mechanics calculations[18] we find that a grid size that results in five to eight points per occupied cell gives the best trade-off between accuracy and performance.

From Table V and other test cases (data not shown) it seems that a density of 2–4 points/Å^2 leads to calculated energies very close to those obtained with higher point densities. For speed, we would suggest using a density of 2 points/Å^2, which leads to deviations in energies that are within 2% higher point density values. In practice, this is a reasonable level of accuracy especially when computing relative changes in solvation free energies rather than absolute free energies. At this point density, a grid size of 1.9 Å is appropriate.

For all previous tests, we have used a convergence tolerance of 10^{-5}. Recall that convergence is reached when the relative change in the calculated reaction field energy is less than this tolerance from one Gauss–Seidel iteration to the next. As seen in the ''Results'' section (Table I), 10^{-5} is a rather stringent tolerance. In production runs, a larger tolerance of 10^{-4} would probably still be acceptable in most cases. This can be seen in the next section where we have results for both convergence tolerances of 10^{-4} and 10^{-5}.

Scaling properties. To assess how the CPU requirements of the method scale with size, reaction field energies were calculated for a series of proteins of increasing size using the parameters just described. In Table VI we summarize the results for a series of increasingly larger proteins: crambin, BPTI, an SH2 domain, and thrombin. We note that the use of 10^{-4} as a convergence tolerance appears to be a good compromise between speed and accuracy and can be used as the convergence tolerance in routine production runs. If we examine the CPU time/Gauss–Seidel iteration, we see, in Figure 2a, that the time/iteration scales sublinearly with respect to number of atoms. This is to be expected because the computational complexity

TABLE VI.
Performance of BEM Calculation vs. Size of Macromolecule.

				CPU time (seconds)				
	Energy	No. of Iterations	Conv. tol.	Calc. σ	E_{norm}	K_{ii}	Setup	Total
Crambin	−121.26	6	10^{-5}	0.77 (0.13)	1.19	0.93	0.64	3.53
	−121.26	6	10^{-4}	0.77 (0.13)	1.19	0.93	0.64	3.53
BPTI	−544.93	42	10^{-5}	7.32 (0.17)	2.18	1.37	0.82	11.69
	−545.22	20	10^{-4}	3.48 (0.17)	2.18	1.37	0.84	7.87
SH2	−656.31	31	10^{-5}	8.77 (0.28)	6.14	2.36	1.25	18.52
	−655.48	19	10^{-4}	5.45 (0.27)	6.14	2.37	1.27	15.23
Papain	−920.20	48	10^{-5}	22.30 (0.46)	19.94	4.21	2.01	48.46
	−920.30	32	10^{-4}	14.73 (0.46)	19.91	4.17	2.02	40.83

Crambin, BPTI, SH2 domain, and papain are proteins with 653, 890, 1617, and 3269 atoms, respectively. Molecular surface points at a density of 2 points/Å^2 were generated using the SIMS program.[23] The grid size used in the multipole calculation was 1.9 Å. The time per Gauss–Seidel iteration is shown in parentheses in the fifth column. The slight difference between the energies for crambin here and those in previous tables is due to the use of a different surface program (SIMS) in this table.

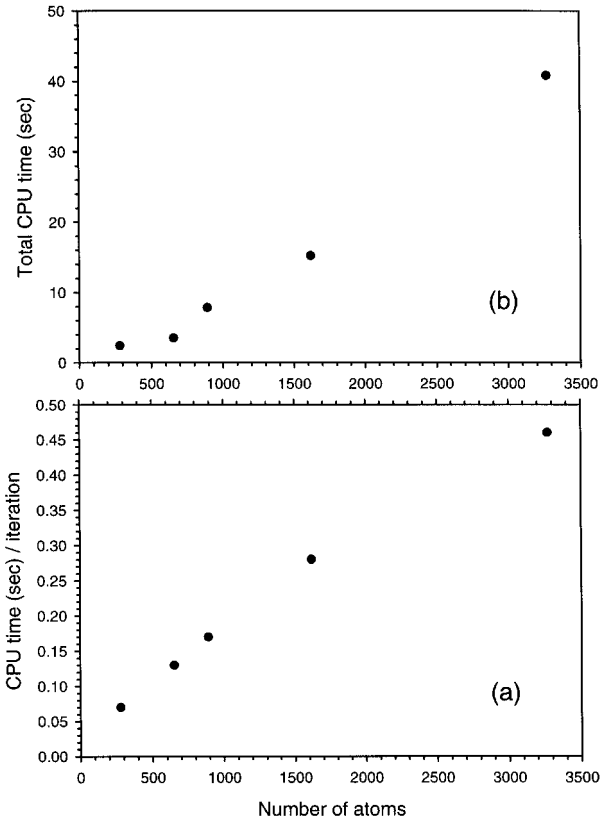

FIGURE 2. CPU requirement vs. molecule size. The plots are based on the data in Tables V and VI.

of the hierarchical multipole calculation is linear with respect to the number of surface points and the number of surface points scales like $N^{2/3}$ with respect to the number of atoms. The number of Gauss–Seidel iterations required differs from protein to protein and is not a simple function of size. These differences in the numbers of iterations partially obscure the precise relationship between the total CPU time and the size of the molecule. Nevertheless, Figure 2b, a plot of the total CPU time versus number of atoms in the molecule, suggests a behavior that appears to be superlinear. The superlinear scaling is probably due to the CPU time for the E_{norm} component (the normal component of the electric field at each surface point due to the solute charges) which is not calculated using a multipole expansion. Despite this superlinear scaling, the total CPU requirements are still quite reasonable even for a protein with over 3000 atoms.

Future improvements. Further improvement in the performance of our algorithm should be possible. As mentioned earlier, the calculation of the electric field of the solute charges at each surface point is currently not treated using a multipole expansion. Because this calculation is only done once at the beginning, we did not invest much effort in speeding it up. However, now that the other parts of the calculation have been sped up dramatically, it is now worthwhile to accelerate this part of the calculation. The work of Vorobjev and Scheraga[13] suggests that a preconditioned conjugate-gradient solution for the solution of a system of equations would converge faster than the Gauss–Seidel method that we currently use. This could possibly improve performance by up to another factor of two.

Conclusion

We have applied hierarchical multipole approximation and fast summation techniques to a boundary element method for use in continuum dielectric calculations. We have systematically studied and identified the minimum level of multipole expansion required for the off-diagonal elements in the boundary element matrices. A fast summation formula has also been derived for the diagonal terms. Using our method, the reaction field energy of a small protein such as BPTI with 890 atoms can be calculated in 7.9 seconds on an ordinary workstation (e.g., SGI R10000). For a larger protein such as papain with 3269 atoms, the calculation takes 41 seconds. Thus, our boundary element method is competitive with and, in the case of smaller proteins, perhaps performs better than the finite-difference approach, which, over the past few years, has itself undergone extensive optimization.

Acknowledgment

The author thanks Hervé Hogues for assistance in C++ programming and Dr. Ling Chan for help on the use of the marching tetrahedra surface program. I also thank Dr. Yury Vorobjev for assistance in using his programs FAMBE and SIMS, which may be downloaded from http://femto.med.unc.edu/FAMBE and http://femto.med.unc.edu/SIMS, respectively.

Appendix

In this Appendix, we describe the multipole expansion and Taylor series expressions used in our calculations. For a more thorough discussion

of the concepts behind the hierarchical multipole and local Taylor expansion method, the reader is referred to detailed descriptions in the literature.[15–17]

Calculation of $\mathbf{K}_F[\sigma]$. At the monopole approximation, the field felt at the center of cell μ from cell ν is [from eq. (10)]

$$\mathbf{E}_\nu(\mathbf{r}_\mu) = q_\nu \frac{\mathbf{r}_\mu - \mathbf{r}_\nu}{r_{\mu\nu}^3} \tag{A1}$$

where q_ν is the total surface charge in cell ν and $\mathbf{r}_\mu$ is the center of cell μ. At some other point in cell μ the field can be approximated to first order using a Taylor expansion about the center of the cell:

$$\mathbf{E}_\nu(\mathbf{r}_i) = q_\nu \left(\frac{\mathbf{r}_\mu - \mathbf{r}_\nu}{r_{\mu\nu}^3} + \mathbf{M}_{\mu\nu}\mathbf{r}_{i\mu} \right) \tag{A2}$$

where $\mathbf{r}_{i\mu} = \mathbf{r}_i - \mathbf{r}_\mu$ and

$$\mathbf{M}_{\mu\nu} = \frac{1}{r_{\mu\nu}^3}\mathbf{I} - \frac{3}{r_{\mu\nu}^5} \begin{pmatrix} x_{\mu\nu}^2 & x_{\mu\nu}y_{\mu\nu} & x_{\mu\nu}z_{\mu\nu} \\ x_{\mu\nu}y_{\mu\nu} & y_{\mu\nu}^2 & y_{\mu\nu}z_{\mu\nu} \\ x_{\mu\nu}z_{\mu\nu} & y_{\mu\nu}z_{\mu\nu} & z_{\mu\nu}^2 \end{pmatrix} \tag{A3}$$

$\mathbf{I}$ is the identity matrix and $x_{\mu\nu}$, $y_{\mu\nu}$ and $z_{\mu\nu}$ are the components of $\mathbf{r}_\mu - \mathbf{r}_\nu$. (Note that the subscript $\mu\nu$ on $\mathbf{M}$ does not indicate matrix elements, but rather indicates that the matrix arises from interactions of cell μ and ν.) If we sum over all the far cells (where far means more than one cell unit away in any direction), the total far field (at this grid level) at any point in cell μ is approximated by

$$\mathbf{E}(\mathbf{r}_i) = \mathbf{a}_\mu + \mathbf{M}_\mu \mathbf{r}_{i\mu} \tag{A4}$$

where

$$\mathbf{a}_\mu = \sum_{\text{far } \nu} q_\nu \frac{\mathbf{r}_\mu - \mathbf{r}_\nu}{r_{\mu\nu}^3} \tag{A5}$$

and

$$\mathbf{M}_\mu = \sum_{\text{far } \nu} q_\nu \mathbf{M}_{\mu\nu} \tag{A6}$$

Eq. (A4) is the basis of the hierarchical multipole calculation. Figure 1 schematically illustrates a multilevel grid of cells. C2, C3, and C4 mark the centers of a particular level 2, 3, and 4 cell, respectively. C3 is a child of C2 and C4 is a child of C3. (We loosely use C2, C3, and C4 to refer to both the center of the cells or the cells themselves depending on the context.) Consider the total far field at some point $\mathbf{r}_i$ in the C4 cell. The electric field contribution from distant cells comes from three levels of distant cells labeled 2, 3, and 4 in Figure 1. (Of course, in a real case there could be more than just three levels of subdivision of the grid.) From eq. (A4), the field at $\mathbf{r}_i$ is given by

$$\mathbf{E}(\mathbf{r}_i) = \mathbf{a}_2 + \mathbf{M}_2\mathbf{r}_{i2} + \mathbf{a}_3 + \mathbf{M}_3\mathbf{r}_{i3} + \mathbf{a}_4 + \mathbf{M}_4\mathbf{r}_{i4} \tag{A7}$$

where the various $\mathbf{a}_k$, $\mathbf{M}_k$ and $\mathbf{r}_{ik}$ values are calculated using the kth-level grid and the appropriate cell center. Using

$$\mathbf{r}_{i2} = \mathbf{r}_{i4} + \mathbf{r}_{43} + \mathbf{r}_{32} \tag{A8}$$

$$\mathbf{r}_{i3} = \mathbf{r}_{i4} + \mathbf{r}_{43} \tag{A9}$$

we can rewrite eq. (A7) as

$$\mathbf{E}(\mathbf{r}_i) = (\mathbf{a}_2 + \mathbf{a}_3 + \mathbf{a}_4 + \mathbf{M}_2(\mathbf{r}_{43} + \mathbf{r}_{32}) + \mathbf{M}_3\mathbf{r}_{43}) + (\mathbf{M}_2 + \mathbf{M}_3 + \mathbf{M}_4)\mathbf{r}_{i4} \tag{A10}$$

This can be written in the same form as eq. (A4)

$$\mathbf{E}(\mathbf{r}_i) = \mathbf{a}_4^{(tot)} + \mathbf{M}_4^{(tot)}\mathbf{r}_{i4} \tag{A11}$$

where $\mathbf{a}_4(tot)$ and $\mathbf{M}_4(tot)$ are obtained from the recursive relations

$$\mathbf{a}_k^{(tot)} = \mathbf{a}_{k-1}^{(tot)} + \mathbf{M}_{k-1}^{(tot)}\mathbf{r}_{k,k-1} + \mathbf{a}_k \tag{A12}$$

$$\mathbf{M}_k^{(tot)} = \mathbf{M}_{k-1}^{(tot)} + \mathbf{M}_k \tag{A13}$$

and initialization conditions

$$\mathbf{a}_2^{(tot)} = \mathbf{a}_2 \tag{A14}$$

$$\mathbf{M}_2^{(tot)} = \mathbf{M}_2 \tag{A15}$$

$\mathbf{r}_{k,k-1}$ is the vector from the center of a parent cell to the center of one of its child cells. Thus, the algorithm involves initially constructing the vector $\mathbf{a}$ and matrix $\mathbf{M}$ for the various level 2 cells. Then, $\mathbf{a}$ and $\mathbf{M}$ are computed for the child cells at the next level using eqs. (A12) and (A13). The process is iterated up to the highest level. Finally, at the highest level cells, $(\mathbf{K}_F[\sigma])_i$ at each individual point, $\mathbf{r}_i$, in the cell is calculated as

$$(\mathbf{K}_F[\sigma])_i = (\mathbf{a} + \mathbf{M}(\mathbf{r}_i - \mathbf{r}_c)) \cdot \mathbf{n}_i \tag{A16}$$

where $\mathbf{a}$ and $\mathbf{M}$ are the accumulated vector and matrix for the cell and $\mathbf{r}_i - \mathbf{r}_c$ is the vector from the center of the cell to point i.

Fast summation of $\mathbf{K}_{ii}$. We now describe a fast summation approximation for K_{ii} as defined in eq. (12). Consider a cell c far from point i. By analogy

with eqs. (A2) and (A3) we can write as a first-order approximation for the contribution to the summation in eq. (12) from points in cell c

$$K_{ii}^{c} \equiv \sum_{j \in \text{cell}} \frac{\mathbf{r}_j - \mathbf{r}_i}{r_{ij}^3} \cdot \mathbf{n}_j A_j = \sum_{j \in \text{cell}} \left(\frac{\mathbf{r}_c - \mathbf{r}_i}{r_{ci}^3} + \mathbf{M}_{ci} \mathbf{r}_{jc} \right) \cdot \mathbf{n}_j A_j \quad \text{(A17)}$$

where $\mathbf{r}_c$ is the center of cell c. We want to rewrite the right-hand side eq. (17) as a set of summations that can be precomputed for each cell

$$K_{ii}^{c} = \frac{\mathbf{r}_{ci}}{r_{ci}^3} \cdot \sum_{j \in \text{cell}} \mathbf{n}_j A_j + \frac{1}{r_{ci}^3} \sum_{j \in \text{cell}} \mathbf{r}_{jc} \cdot \mathbf{n}_j A_j - \frac{3}{r_{ci}^5} \sum_{j \in \text{cell}} (\mathbf{T}_{ci} \mathbf{r}_{jc}) \cdot \mathbf{n}_j A_j \quad \text{(A18)}$$

where $\mathbf{T}_{ci}$ is a matrix of the same form as the matrix in the second term of the right-hand side of eq. (A3). We now define the following quantities

$$\alpha_c = \sum_{j \in \text{cell}} \mathbf{n}_j A_j \quad \text{(A19)}$$

$$\beta_c = \sum_{j \in \text{cell}} \mathbf{r}_{jc} \cdot \mathbf{n}_j A_j \quad \text{(A20)}$$

$$\Gamma_c = \sum_{j \in \text{cell}} [\mathbf{r}_{jc} n_j^x \quad \mathbf{r}_{jc} n_j^y \quad \mathbf{r}_{jc} n_j^z] \quad \text{(A21)}$$

Each of the summations in eqs. (A19)–(A21) can be precomputed for the various cells. Using these precomputed coefficients we can then rewrite eq. (A18) as

$$K_{ii}^{c} = \frac{1}{r_{ci}^3} (\mathbf{r}_{ci} \cdot \alpha_c + \beta_c) - \frac{3}{r_{ci}^5} \mathbf{T}_{ci} \circ \Gamma_c \quad \text{(A22)}$$

where $\mathbf{T}_{ci} \circ \Gamma_c$ is a sum of dot products of the form

$$\mathbf{A} \circ \mathbf{B} \equiv \mathbf{A}_{row1} \cdot \mathbf{B}_{col1} + \mathbf{A}_{row2} \cdot \mathbf{B}_{col2} + \mathbf{A}_{row3} \cdot \mathbf{B}_{col3} \quad \text{(A23)}$$

The construction of the coefficients for the highest level cells is carried out using eqs. (A19)–(A21). However, for the lower level cells we iteratively use the relation between the parent and child cell coefficients

$$\alpha_p = \sum_{\text{child cells}} \alpha_c \quad \text{(A24)}$$

$$\beta_p = \sum_{\text{child cells}} \beta_c + \mathbf{r}_{cp} \alpha_c \quad \text{(A25)}$$

$$\Gamma_p = \sum_{\text{child cells}} (\Gamma_c + [\alpha_c^x \mathbf{r}_{cp} \quad \alpha_c^y \mathbf{r}_{cp} \quad \alpha_c^z \mathbf{r}_{cp}]) \quad \text{(A26)}$$

With all of the coefficients in place, we can then calculate K_{ii} as

$$K_{ii} = 2\pi - \sum_{\text{far cells}} K_{ii}^{c} - \sum_{\text{near } j} \frac{\mathbf{r}_j - \mathbf{r}_i}{r_{ij}^3} \cdot \mathbf{n}_j A_j \quad \text{(A27)}$$

Note that the summation over far cells is carried out hierarchically; that is, starting with the coarsest level cells first and leading up to the highest level cells.[18]

References

1. B. Honig and A. Nicholls, *Science*, **268**, 1144 (1995).
2. B. Honig, K. Sharp, and A.-S. Yang, *J. Phys. Chem.*, **97**, 1101 (1993).
3. M. E. Davis and J. A. McCammon, *Chem. Rev.*, **90**, 509 (1990).
4. K. A. Sharp and B. Honig, *Ann. Rev. Biophys. Biophys. Chem.*, **19**, 301 (1990).
5. J. Warwicker and H. C. Watson, *J. Mol. Biol.*, **157**, 671 (1982).
6. M. K. Gilson, K. A. Sharp, and B. H. Honig, *J. Comput. Chem.*, **9**, 327 (1987).
7. A. Nicholls and B. Honig, *J. Comput. Chem.*, **12**, 435 (1991).
8. R. J. Zauhar and R. S. Morgan, *J. Mol. Biol.*, **186**, 815 (1985).
9. R. J. Zauhar and R. S. Morgan, *J. Comput. Chem.*, **9**, 171 (1988).
10. A. A. Rashin and K. Namboodiri, *J. Phys. Chem.*, **91**, 6003 (1987).
11. R. J. Zauhar and A. Varnek, *J. Comput. Chem.*, **17**, 864 (1996).
12. R. Bharadwaj, A. Windemuth, S. Sridharan, B. Honig, and A. Nicholls, *J. Comput. Chem.*, **16**, 898 (1995).
13. Y. N. Vorobjev and H. A. Scheraga, *J. Comput. Chem.*, **18**, 569 (1997).
14. E. O. Purisima and S. H. Nilar, *J. Comput. Chem.*, **16**, 681 (1995).
15. L. Greengard and V. Rokhlin, *J. Comput. Phys.*, **73**, 325 (1987).
16. J. Shimada, H. Kaneko, and T. Takada, *J. Comput. Chem.*, **15**, 28 (1994).
17. M. O. Fenley, W. K. Olson, K. Chua, and A. H. Boschitsch, *J. Comput. Chem.*, **17**, 976 (1996).
18. H.-Q. Ding, N. Karasawa, and W. A. Goddard III, *J. Chem. Phys.*, **97**, 4309 (1992).
19. J. A. Board Jr., J. W. Causey, J. F. Leathrum Jr., A. Windemuth, and K. Schulten, *Chem. Phys. Lett.*, **198**, 89 (1992).
20. S. J. Weiner, P. A. Kollman, D. T. Nguyen, and D. A. Case, *J. Comput. Chem.*, **7**, 230 (1986).
21. S. L. Chan and E. O. Purisima, *Comput. Graphics*, **22**, 830 (1998).
22. S. L. Chan and E. O. Purisima, *J. Comput. Chem.*, in press.
23. Y. N. Vorobjev and J. Hermans, *Biophys. J.*, **73**, 722 (1997).
24. M. L. Connolly, *J. Appl. Cryst.*, **16**, 548 (1983).